Computational Analysis of a Bead on a Wire

A Senior Project

presented to

the Faculty of the Physics Department

California Polytechnic State University, San Luis Obispo

In Partial Fulfillment

of the Requirements for the Degree

Bachelors of Science

by

Kristofer Schobert

December, 2013

Table of Contents

I. Introduction

In elementary physics courses, one commonly comes across the problem of an object sliding down a frictionless incline. To analyze the object's motion, one refers to a free-body diagram of the forces acting on the object, and then applies the kinematic equations to solve for the object's position, velocity, and acceleration. This approach works beautifully for an object moving along a straight line (e.g. an inclined plane); however, if the object is confined to a curved line, this approach is no longer valid.

This is because the acceleration of the object is not constant when the slope of the line is not constant. The kinematic equations rely on a constant acceleration, thus they are no longer applicable. When attempting to analyze this more complex problem one encounters non-linear equations of motion. In many cases, these equations cannot be solved analytically, so one must approach the problem differently.

We used a computational procedure, Euler's method, to solve these non-linear equations. Euler's method was chosen because the technique is rather simple, and we are ultimately focused on the physics of our systems.[1] The equation for Euler's method,

$$y_{n+1} = y_n + f(y_n, t_n)h = y_n + \left(\frac{dy}{dt}\right)_n \Delta t \tag{1}$$

where $y(t)$ is the position of the particle, and h is the interval. This describes the method for moving from step n to step n+1.[1] As shown we will be concerned with the cases where $f(y_n,t_n) = dy_n/dt_n$. To implement this method, we compute the acceleration, velocity, and position of the object at a specific time. Assuming the object's acceleration does not change significantly over a short time interval, the kinematic equations are used to compute the object's trajectory for this short time interval. After calculating the new acceleration, velocity, and position of the object, one repeats this process. As long as the time interval is kept sufficiently small, one can calculate the trajectory of the object very accurately over long periods of time.

The computer programming environment we chose to perform this procedure and animate our object's trajectory, is VPython. This programming software requires only basic coding to create extremely useful and informative animations. A convenient feature of VPython is its treatment of a vector. VPython understands certain arrays are 3-dimensional vectors, thus one does not have to incorporate three different equations, one for each component of an array. Therefore, VPython is extremely useful for representing physical systems. One can find all of the VPython code we have written in the Appendix.

Our treatment of these problems closely follows the approach of Bensky and Moelter.[2] However, our treatment expands their work by introducing alternative forces.

II. Object Subjected to the Gravitational Force

A. Motion Confined to a Non-parameterized Curve

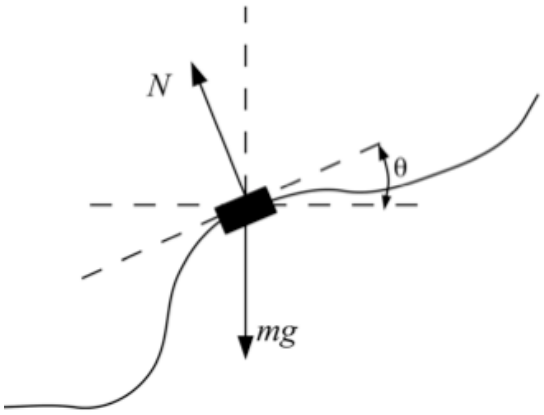

Figure 1: Free-body diagram of object constrained to curve.

When analyzing the motion of any object through space, all one needs is initial conditions and a function describing the acceleration of the object to determine the entire trajectory of an object. Once one knows the acceleration of an object he or she can integrate the acceleration once or twice to determine a function describing the velocity or position of that object. We will proceed with the analysis to find the acceleration of an object subject to gravity and confined to a frictionless curve described by a function.

To begin our analysis of the system, we examine the free-body diagram, shown in Figure 1. We have chosen our coordinate axes so that the y-axis is antiparallel to the object's weight, $m\boldsymbol{g}$. The function which describes the curve will be called *y(x)*. The angle, θ, represents the angle between the positive x-axis and the tangent of the curve, *y'*. Also, one will notice the normal force vector, $\boldsymbol{N}$, is perpendicular to the tangent of the curve. We now apply Newton's second law to obtain

$$\sum F_x = N_x = -N\sin\theta = ma_x \tag{2}$$

and

$$\sum F_y = N_y - mg = N\cos\theta - mg = ma_y \tag{3}$$

We seek the components of acceleration, but have two equations with three unknowns, N, a_x, and a_y. To eliminate N we divide Eq. (1) by (2), giving

$$a_x = -(a_y + g)\tan\theta = -(a_y + g)y' \tag{4}$$

where $\tan\theta = dy/dx = y'$. We now have an equation describing the relationship between the two components of acceleration. Once we determine an equation for either a_x or a_y, in terms of other known variables, we will be able to determine equations for both

components of acceleration. To do so we will need to analyze the effects of constraining the object to a curve.

The constraint dictates that the components of position, velocity, and acceleration are not independent of each other. Therefore, they can be expressed in terms of each other. For instance,

$$v_y = \frac{dy}{dt} = \frac{dy}{dx}\frac{dx}{dt} = y' v_x \tag{5}$$

and

$$a_y = y'' v_x^2 + y' a_x . \tag{6}$$

Here we have employed the chain rule and the definitions $v_x = dx/dt$ and $a_x = dv_x/dt$. By substituting Eq. (5) into Eq. (3), one finds

$$a_x = \frac{-(y'' v_x^2 + g) y'}{1 + y'^2} . \tag{7}$$

Finally, we have arrived at an equation for a component of the object's acceleration. By substituting Eq. (6) into Eq. (3) one will have determined equations for both components of acceleration. Therefore, one can determine the object's entire trajectory as it travels along the curve.

We proceed by implementing the Euler method. Given an initial position and velocity, one can find the initial acceleration via equations (6) and (7). Assuming the acceleration does not change over a small amount of time, one can step forward in time and find the resulting velocity and position.

$$v_{n+1} = v_n + a_n \Delta t \tag{8}$$

$$x_{n+1} = x_n + v_n \Delta t \tag{9}$$

From here on we will often refer to the object as a bead and the curve as a wire. The systems we are analyzing are often known in the physics community as "a bead on a wire" systems, hence our adopted language.

We have chosen the function $y(x) = \tanh(x) + 1$ for the wire, shown in Figure 2. This function is reminiscent of a simple incline. It has horizontal sections at the bottom and top of the incline and the transitions between these sections are continuous.

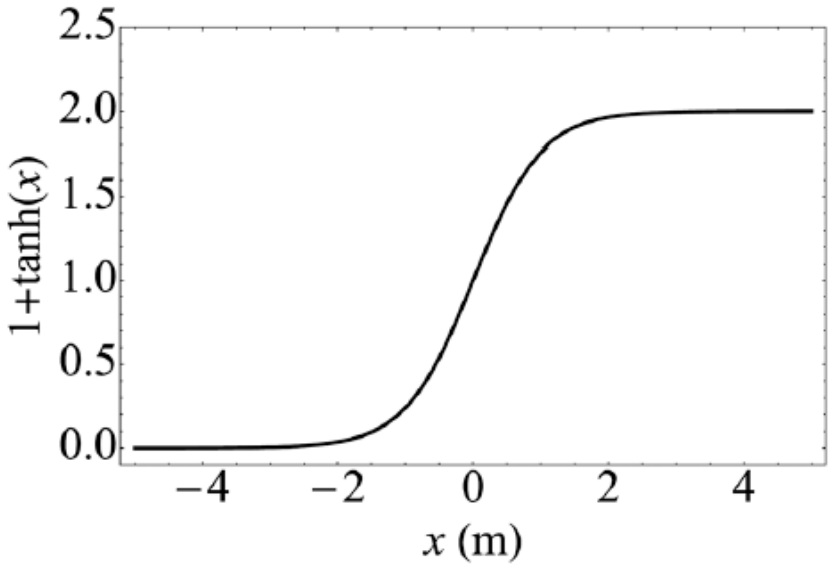

Figure 2: A plot of tanh(x) + 1. The smooth transitions between the horizontal sections and the inclined section make this a suitable function for our purposes. The "(m)" is arbitrary.

To test that the program is running properly, we have chosen to inspect the conservation of energy. The mechanical energy of the bead should be conserved. Thus the kinetic energy of the bead at the beginning should equal the kinetic energy and gravitational potential energy of the bead at the end of the wire. Given a time step size of 0.001, a mass of 1, and an initial velocity of 10, the initial mechanical energy is 50, while the final mechanical energy is 50.0149. (Note: the acceleration due to gravity is chosen to be 9.8. That will remain the same throughout our entire treatment.) The value of the initial mechanical energy was calculated directly from the givens, whereas the final mechanical energy was calculated via the stepping through the Euler method. The small discrepancy of 0.030% from the initial value is due to the inaccuracy of the Euler method. If the step size is made smaller the accuracy will increase. If a time step of 0.0001 is used, the final mechanical energy is 50.0030. This result is even closer to the initial energy; the percent error is only 0.0060%.

Interesting features arise in the position and velocity plots shown in Figure 3 and Figure 4. The position in the y-direction (the blue line) looks very similar to the shape of the wire. This is because with an initial velocity of 6.4, the bead glides up the incline and continues moving with a velocity, which is smaller yet similar to its initial velocity. This slowing of the bead results in the continuously increasing stretching of the original tanh(x) shape as time increases. The x-position steadily increases until $t \approx 0.6$, the time at which it begins to climb the incline. At this time the slope of the red line decreases, signaling its decrease in velocity. Once the bead has arrived at the higher horizontal section at $t \approx 1.4$, its velocity stays constant, hence the consistent slope from then on.

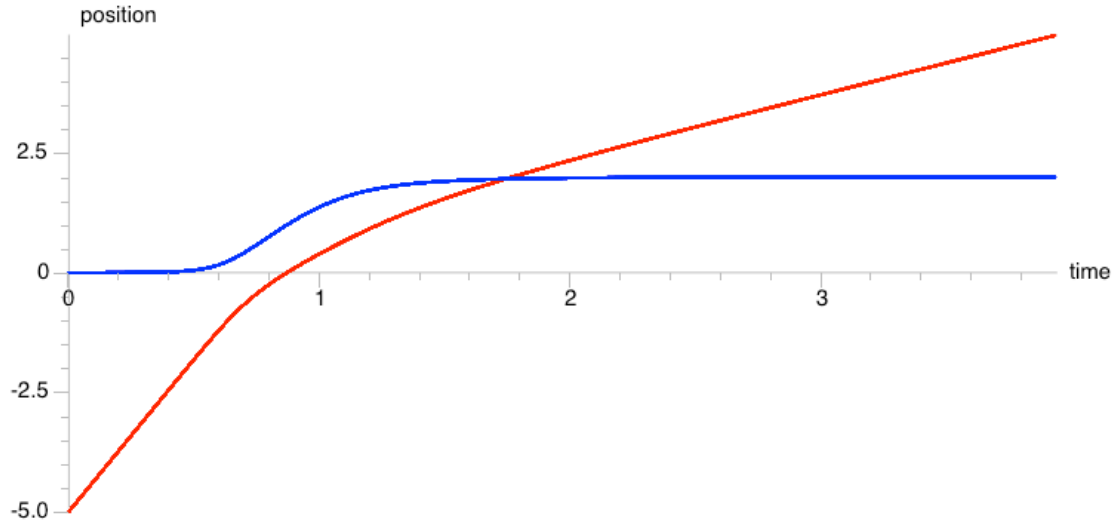

Figure 3: The x and y position of a bead subjected to gravity versus time. The bead is traversing a wire described by the function tanh(x) + 1. The bead has a mass of 1 and an initial velocity of 6.4. The position in the x-direction is described by the red line, while the position in the y-direction is shown by the blue line.

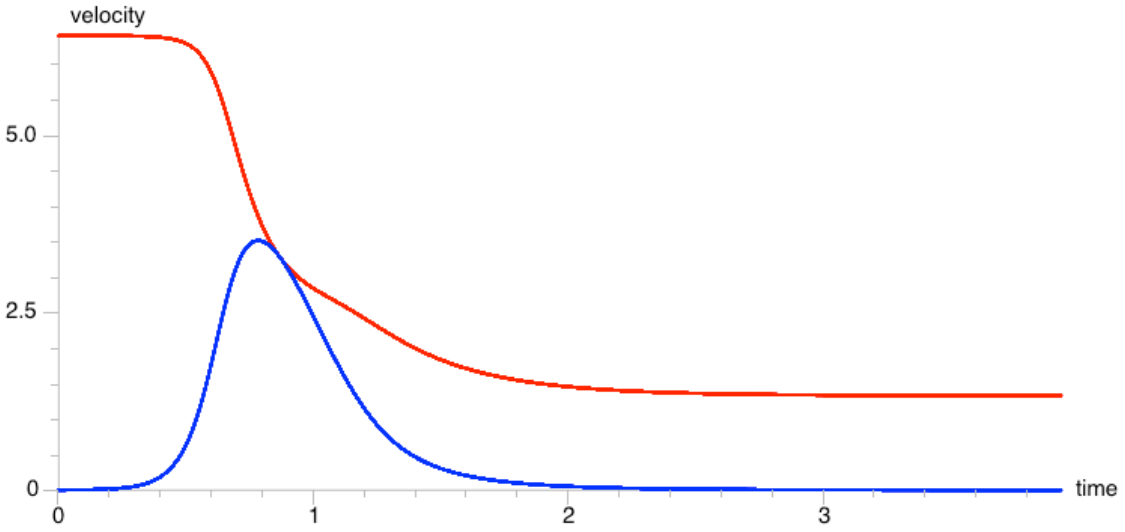

Figure 4: The x and y velocity of a bead subjected to gravity versus time. The bead is traversing a wire described by the function tanh(x) + 1. The bead has a mass of 1 and an initial velocity of 6.4. The velocity in the x-direction is described by the red line, while the velocity in the y-direction is shown by the blue line.

The plot of the velocity components versus time also has some curious features. One notable property is how both components of velocity are equal at $t \approx 0.8$. This is because the slope of the wire is 45˚ with respect to the horizontal at $x = 0$. The derivative of the wire with respect to x is $\text{sech}^2(x)$. This leads us to the fact that $\text{sech}^2(0) = 1$. A slope of one indicates a small displacement in the x-direction is equal to the displacement in the y-direction, and thus the velocities in both directions are equal in magnitude.

A more discrete yet puzzling attribute to the velocity plot is the small hump on the red x-component line at the time $t = 1.2$. This small increase in acceleration occurs because the bead is confined to a wire. As the bead traverses the turn at the top of the incline, the shape of the wire transfers the object's velocity into the x-direction while the force of gravity acts as a retarding force. This hump occurs because as these two influences on motion are not balanced over the course of this turning. The wire directing the bead into the x-direction is at first more dominant than the force of gravity, but afterwards the converse is true.

B. Motion Confined to a Parameterized Curve

If one desires to analyze a situation where a bead is constrained to a wire with a loop then a single-valued function will not suffice in describing the shape of the wire. A parameterized curve is required. Assuming the wire is confined to a plane, the curve can be parameterized by a variable u, so that $x = x(u)$ and $y = y(u)$. Given this parameterization, the components of velocity are

$$v_x = \frac{dx}{dt} = \frac{dx}{du}\frac{du}{dt} = x'\dot{u} \tag{10}$$

and

$$v_y = \frac{dy}{dt} = \frac{dy}{du}\frac{du}{dt} y'\dot{u}. \tag{11}$$

The components of acceleration are

$$a_x = \frac{d^2x}{du^2}\left(\frac{du}{dt}\right)^2 + \frac{dx}{du}\frac{d^2u}{dt^2} = x''\dot{u} + x'\ddot{u} \tag{12}$$

and

$$a_y = \frac{d^2y}{du^2}\left(\frac{du}{dt}\right)^2 + \frac{dy}{du}\frac{d^2u}{dt^2} = y''\dot{u} + y'\ddot{u}. \tag{13}$$

In this notation, a dot denotes a derivative with respect to time, while primes denote a derivative with respect to u. By following steps similar to those in the beginning of Section IIA, we find that

$$\ddot{u} = \frac{-\dot{u}(x'x''+y'y'') - gy'}{(x')^2 + (y')^2}. \tag{14}$$

Using the Euler method in this case is slightly different than in the single-valued function case. We must apply the Euler method to the variable u which is similar to what we did with position and velocity as shown in equations (8) and (9). Given an initial u

and $\dot{u}$ we can find the resulting u and $\dot{u}$ by stepping forward in time once the initial $\ddot{u}$ is found via equation (12). The Euler equations governing this process are

$$\dot{u}_{n+1} = \dot{u}_n + \ddot{u}_n \Delta t \tag{15}$$

and

$$u_{n+1} = u_n + \dot{u}_n \Delta t \ . \tag{16}$$

The Euler equations governing the relationships between acceleration, velocity, and position for the parameter u are the same as in the case of the single-valued function curve.

We have chosen the Trisectrix of Maclaurin as our parametric curve. This shape was appropriate because it is a simple loop much like one found on a rollercoaster. The curve can be described by the equations

$$x(u) = \frac{bu(u^2 - 3)}{u^2 + 1} \tag{17}$$

and

$$y(u) = \frac{-bu(u^2 - 3)}{u^2 + 1}. \tag{18}$$

An example of this curve is shown in Figure 5. In this example $b = 0.1$, and $-5 \le u \le 5$. For our treatment we have chosen $b = 3$, and $-2\pi \le u \le 2\pi$.

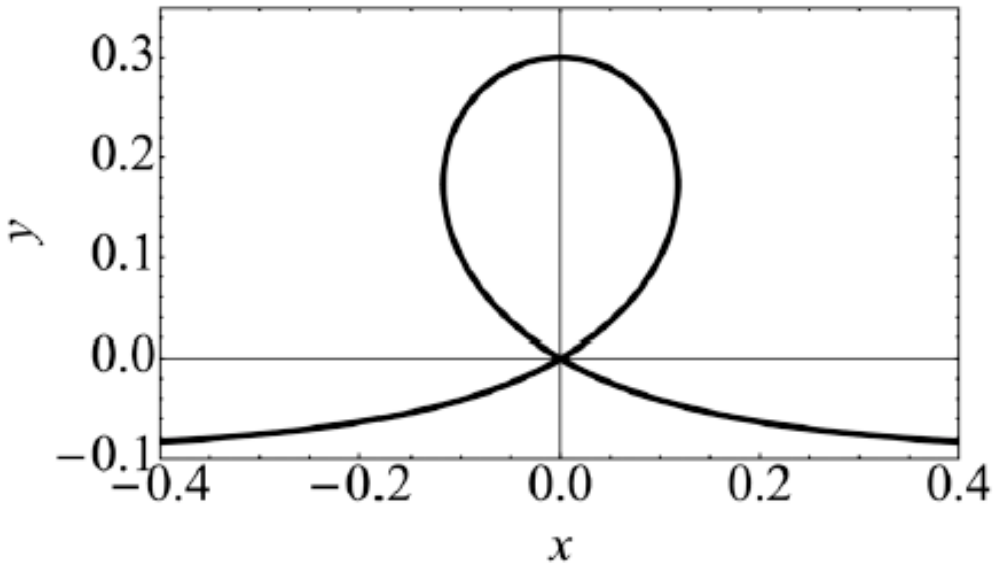

Figure 5: The Trisectrix of Maclaurin for $b = 0.1$ and $-5 \le u \le 5$.

We have chosen to inspect conservation of energy to check that our code is running properly. The total mechanical energy at the beginning of the curve should be the same value as that at the top of the curve. For an initial velocity in the x-direction of 20 and a time step of 0.001, the initial mechanical energy is 200.157 and the mechanical energy at the top of the loop is 200.240. This small discrepancy is due to the approximations made using the Euler method. This method has left us with a 0.041% error - an acceptable amount for our treatment.

Interesting features arise in the position and velocity plots shown in Figure 6 and Figure 7. In this situation, the bead traverses the loop quickly so that the normal force's direction is always toward the center of the loop. In the position versus time plot, the *y*-position of the bead appears as a nice symmetric curve. This is due to both the symmetry of the wire along the line $x = 0$ and the symmetry of the forces acting on the bead. Also, some interesting features occur in both plots when the bead is at the very top of the loop. This occurs at the time $t \approx 1.05$. At this location, the maximum *y*-position occurs while the *x*-position is equal to zero. The velocity plots show that there is a minimum in the *x*-velocity at the top of the loop. This occurs because the bead only moves in the negative *x*-direction at the top of the loop and, it is directed solely in the x-direction at the highest point of the loop. The *y*-velocity is zero at this time because the tangent line at the very top of the loop is zero.

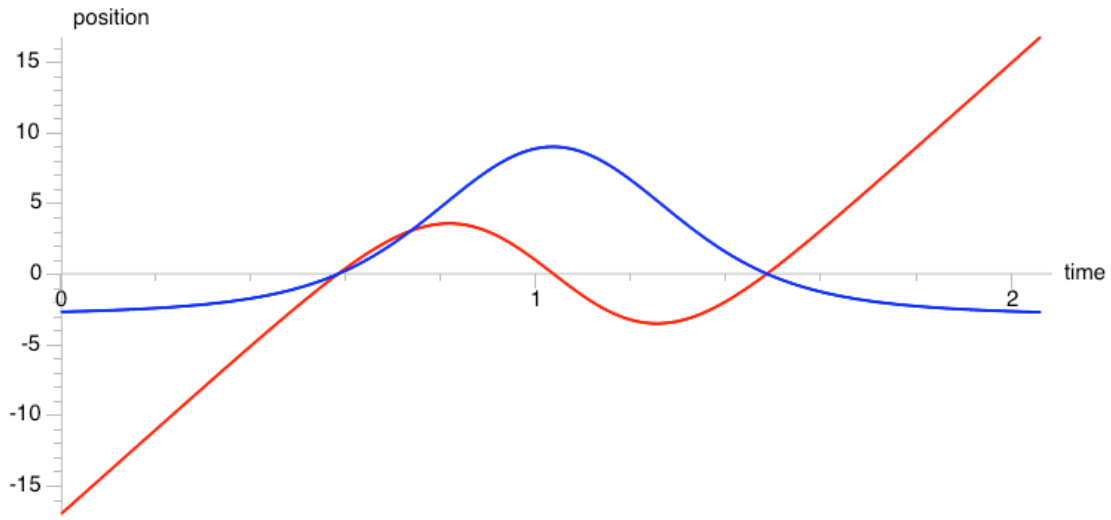

Figure 6: The x and y position of a bead subjected to gravity versus time. The bead is traversing a wire with the shape of a Trisectrix of Maclaurin. The bead has a mass of 1 and an initial velocity in the x-direction of 30. The position in the x-direction is described by the red line, while the position in the y-direction is shown by the blue line.

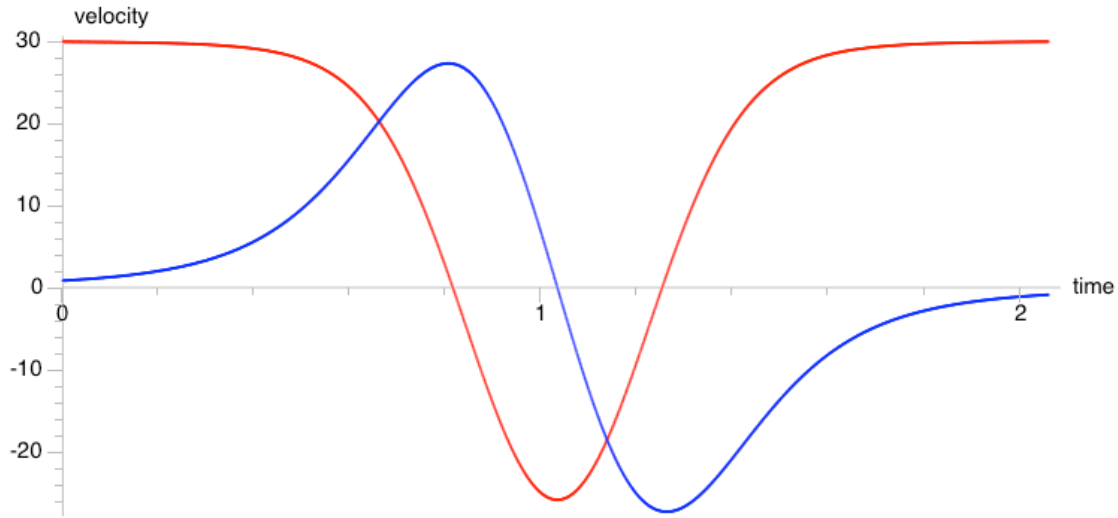

Figure 7: The x and y velocity of a bead subjected to gravity versus time. The bead is traversing a wire with the shape of a Trisectrix of Maclaurin. The bead has a mass of 1 and an initial velocity in the x-direction of 30. The velocity in the x-direction is described by the red line, while the velocity in the y-direction is shown by the blue line.

Another phenomenon arises in the velocity vs. time plots. Excluding the very beginning and end of the plot, each time one component of the bead's velocity reaches a maxima or minima the other component's velocity is zero. This is due to the geometry of the Trisectrix of Maclaurin. At $t \approx 0.65$ and $t \approx 1.25$, the y-component of the bead's velocity is at an extrema, and the x-component of the bead's velocity is zero. At these times, the bead is moving completely vertically in the loop. Also at $t \approx 1.05$, the y-velocity is zero while the x-velocity is at an extrema. This happens when the bead is at the very top of the loop.

The next three figures, Figure 8, Figure 9, and Figure 10, show the same situation as in Figures 6 and 7, but the initial velocity is smaller. With an initial velocity of 15.5, the bead has just enough mechanical energy to make it to the top of the loop and traverse the top of the loop to descend down the opposite side.

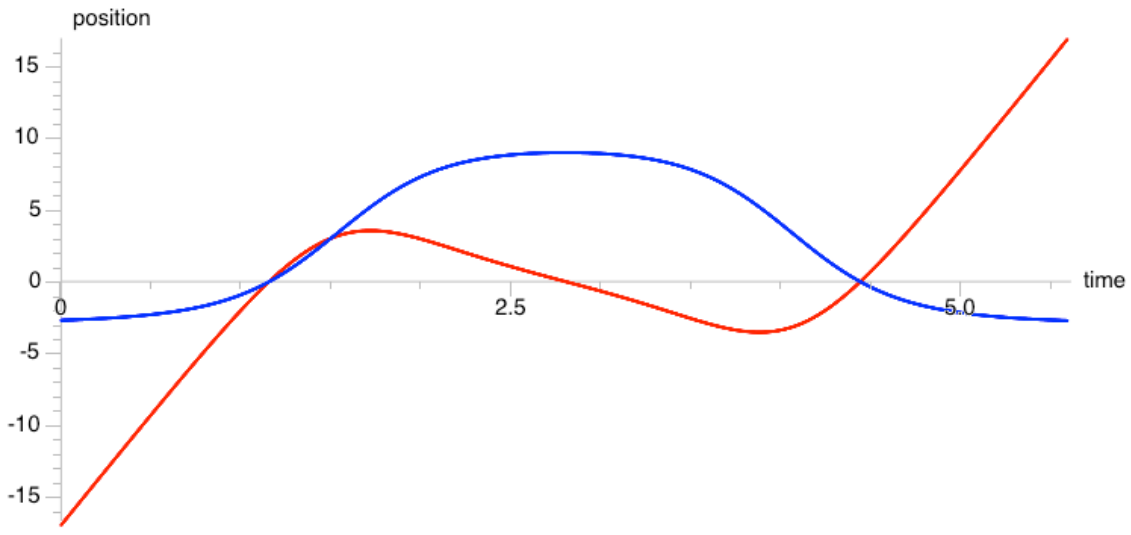

Figure 8: The x and y position of a bead subjected to gravity versus time. The bead is traversing a wire with the shape of a Trisectrix of Maclaurin. The bead has a mass of 1 and an initial velocity in the x-direction is of 15.5. The position in the x-direction is described by the red line, while the position in the y-direction is shown by the blue line.

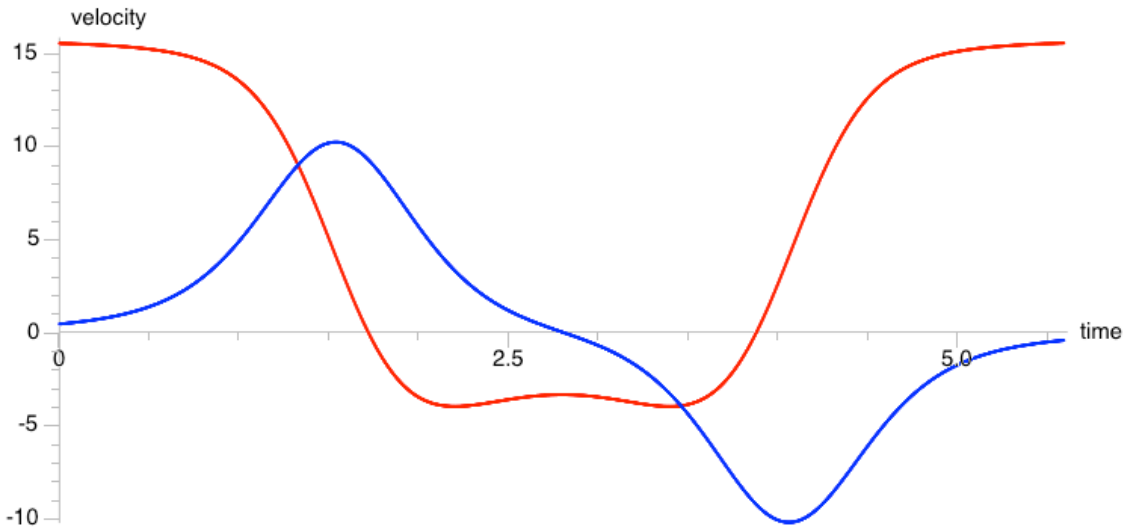

Figure 9: The x and y velocity of a bead subjected to gravity versus time. The bead is traversing a wire with the shape of a Trisectrix of Maclaurin. The bead has a mass of 1 and an initial velocity in the x-direction of 15.5. The velocity in the x-direction is described by the red line, while the velocity in the y-direction is shown by the blue line.

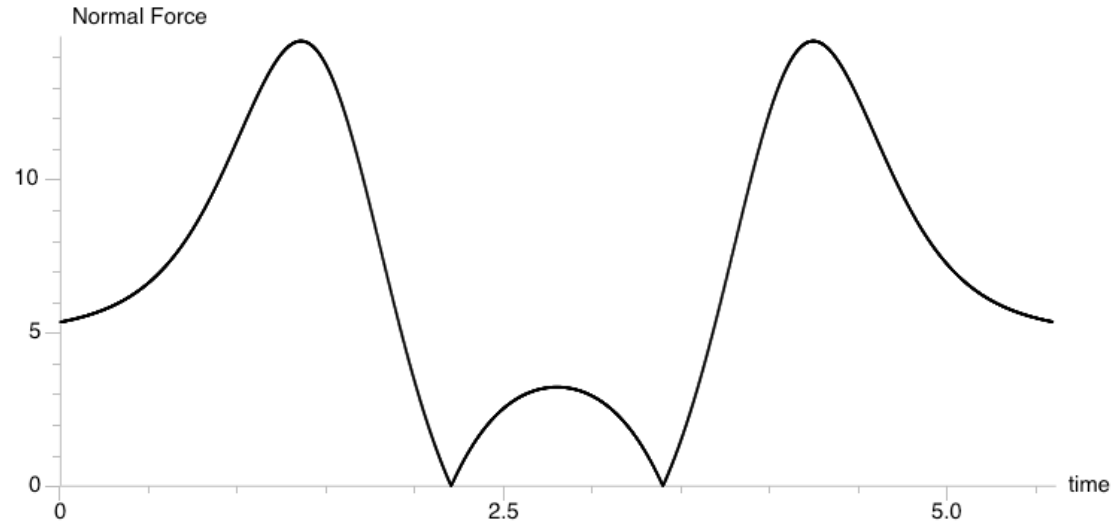

Figure 10: The mangnitude of the normal force of a bead subjected to gravity versus time. The bead is traversing a wire with the shape of a Trisectrix of Maclaurin. The bead has a mass of 1 and an initial velocity in the x-direction of 15.5.

The intriguing nature of this situation is displayed well in the plot of the magnitude of the normal force versus time (Figure 10). The bead begins its trajectory by

climbing the loop. During this process, the normal force is directed antiparallel to the curve and towards the inside of the loop. At $t \approx 2.2$, the normal force switches from pointing towards the inside of the loop to pointing towards the outside of the loop. From $t \approx 2.2$ to $t \approx 3.4$, the bead slides along the top of the loop, resting on the wire. Finally at $t \approx 3.4$, the normal force switches to pointing towards the inside of the loop and the bead traversing the remaining portion of the wire.

The plots of position versus time and velocity versus time are far more complex than the plots for an initial x-velocity of 30. However, one will notice that the symmetry of the bead's trajectory still exists in this more complex situation. Both components of position and velocity are symmetric/antisymmetric about the time $t \approx 2.8$. This is because the only force doing work on the bead along the wire is the object's weight, which is a conservative force. The other force, the normal force, is always antiparallel to the bead's displacement along the wire, thus it does not affect the bead's velocity along the wire. Both the symmetry of the shape of the loop and symmetry of the work done by the bead's weight are cause the elements object's motion to be symmetric about the time when the bead is at the top of the loop, $t \approx 2.8$.

III. An Object Subjected to Alternative Forces

A. Electric and Magnetic Forces

We have chosen to discuss the bead subjected to the electric, magnetic and gravitational forces, while confined to a curve described by the Trisectrix of Maclaurin. The VPython code for the same forces acting on a bead constrained to a wire described by the hyperbolic tangent function is available in the Appendix. To obtain the expression for $\ddot{u}$ in this system, one should follow the steps similar to those in Section II.

We have chosen to analyze a system where the bead has a mass of 1, a charge of 1, and an initial x-velocity of 38. The magnetic field strength is 1, and the electric field strength is 30.

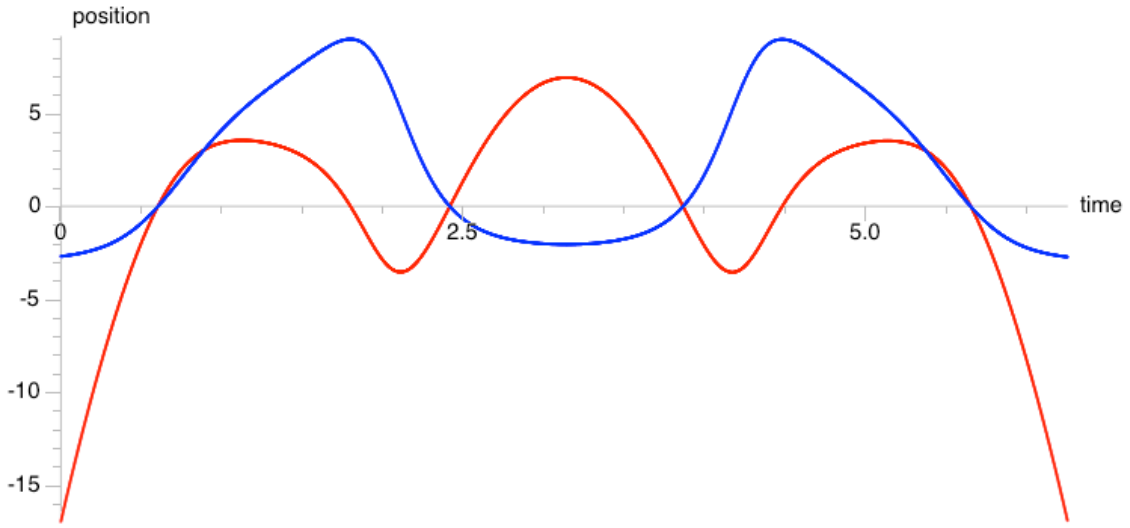

Figure 11: The x and y position of a bead, subjected to gravity, an electric field, and a magnetic field, versus time. The bead is traversing a wire with the shape of a Trisectrix of Maclaurin. The

bead has a mass of 1, a charge of 1, and an initial velocity in the x-direction of 38. The electric field strength is 1, and the electric field strength is 30. The position in the x-direction is described by the red line, while the position in the y-direction is shown by the blue line.

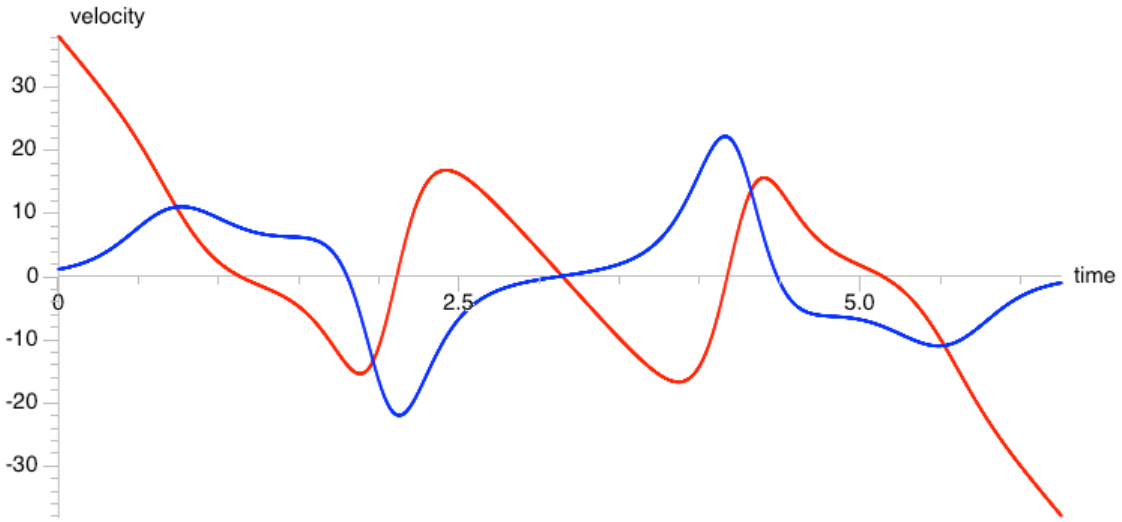

Figure 12: The x and y velocity of a bead, subjected to gravity, an electric field, and a magnetic field, versus time. The bead is traversing a wire with the shape of a Trisectrix of Maclaurin. The bead has a mass of 1, a charge of 1, and an initial velocity in the x-direction of 38. The electric field strength is 1, and the electric field strength is 30. The velocity in the x-direction is described by the red line, while the velocity in the y-direction is shown by the blue line.

As one can easily see from Figures 11 and 12, the addition of the electric force and magnetic force yields a much more complex system. In this trajectory, the bead traverses the loop, comes rest, and then traverses the loop backward and finishes at its starting position. The electric force, which points in the negative x-direction, is responsible for the bead's reversing motion.

From Figures 11 and 12 one sees either symmetry or antisymmetry of the lines about $t = 3.1$. This is because all of the forces doing work on the bead, the force of gravity and the electric force, are conservative forces. The magnetic force is always perpendicular to the bead's motion. Thus, it does no work on the bead nor does it affect the bead's trajectory. To demonstrate this, we compare the times for the bead to return to its starting position in for two very different magnetic field strengths. With a magnetic field of 1 unit is applied to the bead, the time required for the bead to return to its starting position is 6.253. With a magnetic field of 1000 units, the time required for the bead to its initial position is again 6.253. Although the magnetic fields varied by three orders of magnitude, the systems yield the same final time, and the latter system produces plots identical to Figures 11 and 12. It is very apparent that the magnetic force has no effect on the object trajectory.

If one wished to investigate the trajectory of the bead subjected to the same forces but on a wire described by a tanh function, consult the appendix.

B. Kinetic and Static Friction

We have also chosen to analyze the situation in which the bead is subjected to kinetic and static friction and its weight while constrained to the Trisectrix of Maclaurin. One can view the position and velocity plots of this situation in Figure 13 and 14. In this treatment the bead is given a mass of 1, a coefficient of kinetic friction of 0.8, a coefficient of static friction of 13, and an initial velocity in the x-direction of 65.64.

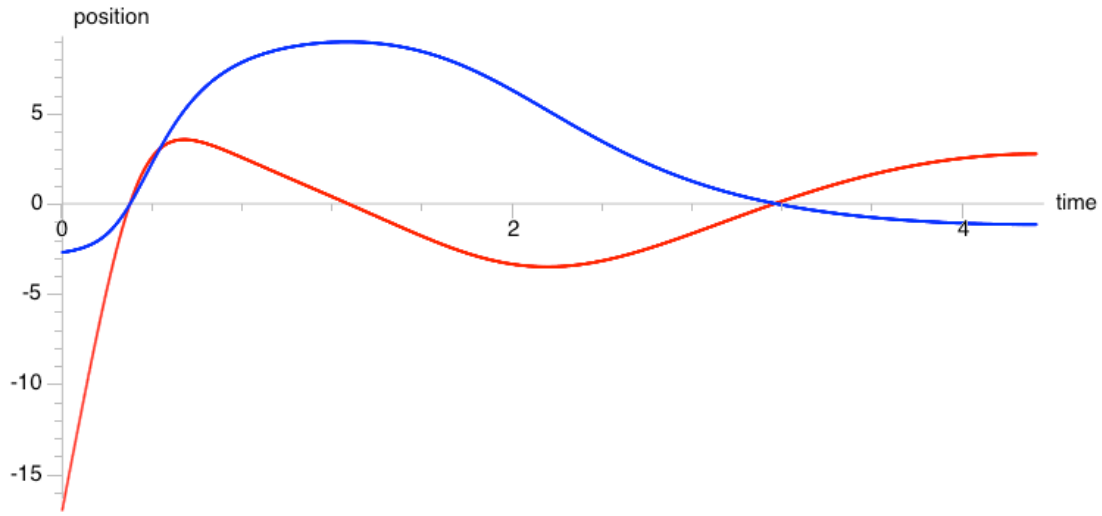

Figure 13: The x and y position of a bead, subjected to gravity and friction versus time. The bead is traversing a wire with the shape of a Trisectrix of Maclaurin. The bead has a mass of 1, a coefficient of kinetic friction of 0.8, a coefficient of static friction of 13, and an initial velocity in the x-direction of 65.64. The position in the x-direction is described by the red line, while the position in the y-direction is shown by the blue line.

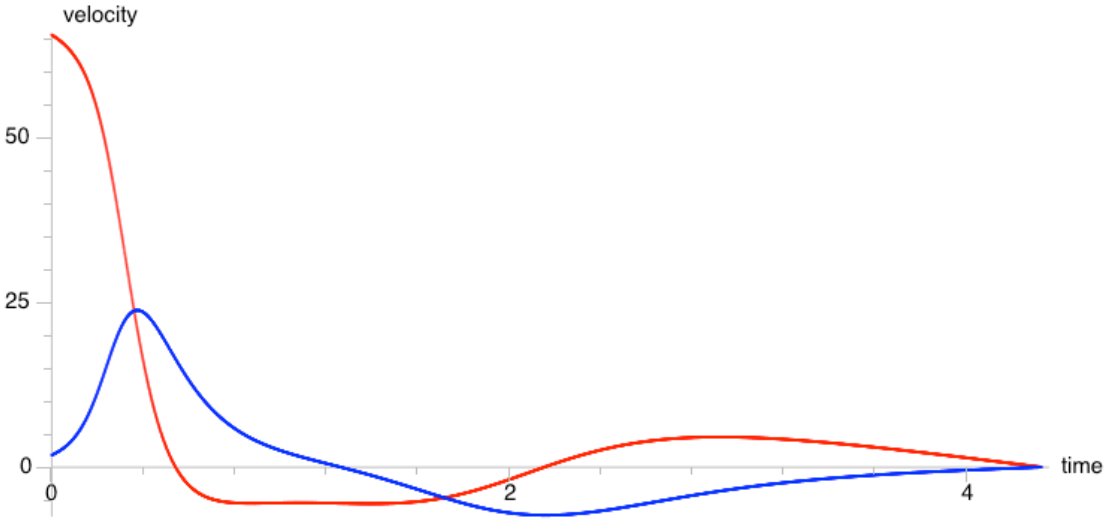

Figure 14: The x and y velocity of a bead, subjected to gravity and friction versus time. The bead is traversing a wire with the shape of a Trisectrix of Maclaurin. The bead has a mass of 1, a coefficient of kinetic friction of 0.8, a coefficient of static friction of 13, and an initial velocity in

the x-direction of 65.64. The velocity in the x-direction is described by the red line, while the velocity in the y-direction is shown by the blue line.

As one can see from these two plots, the force of friction is not a conservative force. If friction were a conservative force, there would be either symmetry or antisymmetry of these curves about some point in time.

Another interesting feature is how the slope of the velocity plots in Figure 14 are generally greater than the slopes after $t \approx 1.3$, the time during which the bead traverses the top of the loop. This is because on the way up the loop ($t < 1.3$), friction and the bead's weight, the two forces doing work on the bead, are pointing in similar directions. The bead's weight is always completely in the negative-y direction, while the y-component of friction is also negative during this time. This leads to a greater acceleration and thus the great slopes of the velocity versus time plots. After the bead traverses the top of the loops and begins to travel downward friction and gravity begin to oppose each other. During this interval when $t > 1.3$, the y-component of friction is always positive. Thus the two forces doing work on the bead are generally opposing each other and thus the bead accelerates less and the slope of the velocity versus time plots are smaller.

If one analyzes a bead constrained to a curve described by the function $y = \tanh(x) + 1$, one can observe a very interesting phenomenon of friction. The position and velocity plots for this system can be seen in Figures 15 and 16.

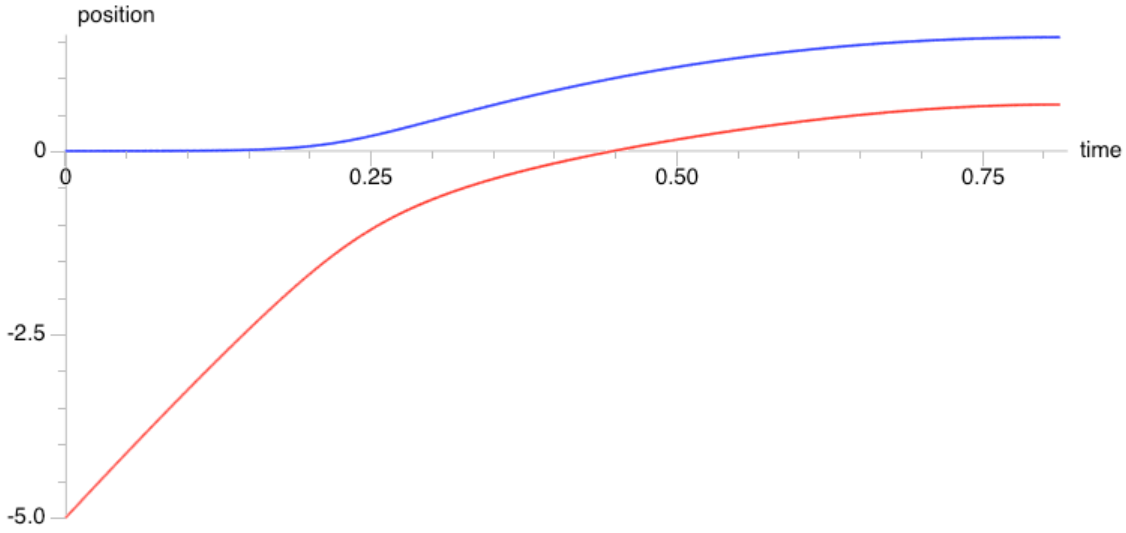

Figure 15: The x and y position of a bead, subjected to gravity and friction versus time. The bead is traversing a wire with the shape of $y = \tanh(x) + 1$. The bead has a mass of 1, a coefficient of kinetic friction of 1, a coefficient of static friction of 10, and an initial velocity of 18. The position in the x-direction is described by the red line, while the position in the y-direction is shown by the blue line.

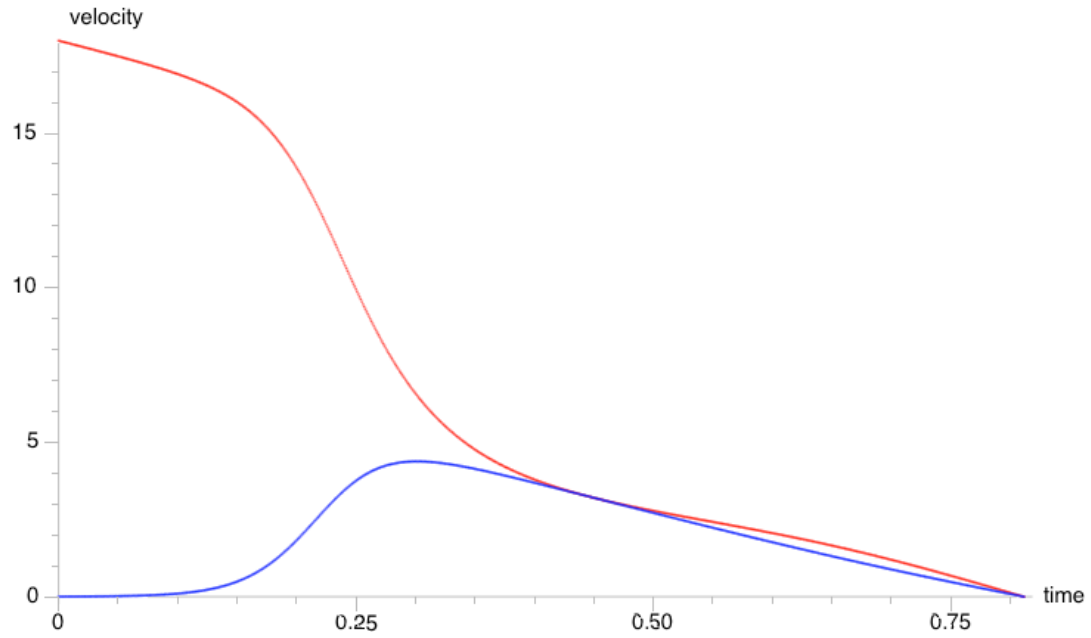

Figure 16: The x and y position of a bead, subjected to gravity and friction versus time. The bead is traversing a wire with the shape of $y = \tanh(x) + 1$. The bead has a mass of 1, a coefficient of kinetic friction of 1, a coefficient of static friction of 10, and an initial velocity of 18. The velocity in the x-direction is described by the red line, while the velocity in the y-direction is shown by the blue line.

In this system, the bead climbs the incline of the hyperbolic tangent function until it comes to rest at the time $t \approx 0.81$. At this critical point in time, the bead can either slide back down the incline or stay at its location indefinitely. In our system, the bead stops and does not continue down the incline, thus the latter is true. This occurs because static friction takes hold at this time. Whenever an object is subjected to kinetic friction and comes to rest, it may or may not stop at that position because static friction may or may not be great enough to hold the object. In our case, the product of the coefficient of static friction and the normal force was great enough to do so. If we had chosen a small enough coefficient of static friction, the object would have continued to slide back down the incline after $t \approx 0.81$.

IV. Conclusion

Through this treatment of a bead constrained to a wire, many interesting phenomenon can be observed. We have discussed a few interesting features of a couple systems, but many more exist within these systems. If one alters one parameter in any of the VPython code we have created, which can be found in the Appendix, one may come across some very interesting results.

The VPython code we have created has been written so one can easily use any function or parameterized curve to described the wire that the bead is confined too. All one has to do is simply change the first few lines of our code where we describe the shape of the wire by either the hyperbolic tangent function or the Trisectix of Maclaurin.

If one would like to expand on this project, we have thought of a few different ways to do so. One could look subject the bead to different forces, for instance linear and quadratic air drag. Also, it would be a challenging yet fulfilling exercise to create a VPython code, which describes a bead constrained to a wire that protrudes into the third dimension. Thus, one could analyze the forces that act on a bead constrained to a helix for instance.

V. References

1. Weisstein, Eric W. "Euler Forward Method." From *MathWorld*--A Wolfram Web Resource. http://mathworld.wolfram.com/EulerForwardMethod.html

2. Thomas J. Bensky and Matthew J. Moelter "Computational problems in introductory physics: Lessons from a bead on a wire," Am. J. Phys. 81, 165 (2013); doi: 10.1119/1.4773561

VI. Appendix

A. Image of VPython Animation

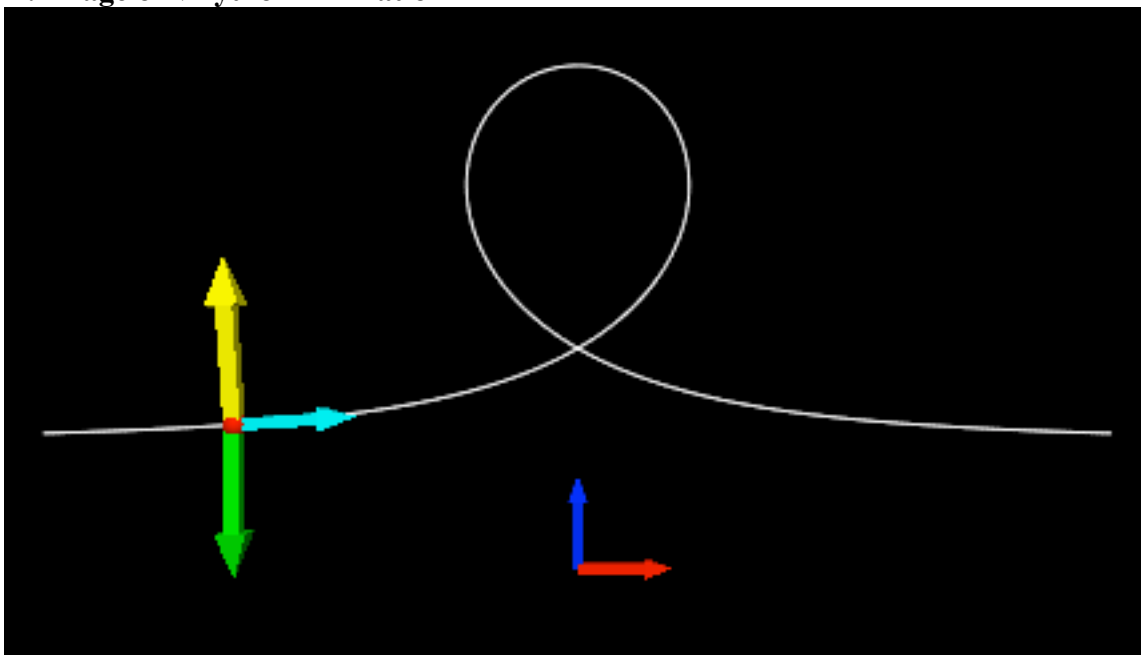

Figure 17: A snapshot of the animation of a bead subjected to gravity confined to a curve described by the Trisectrix of Maclaurin. The bead is shown as a red sphere. The force of gravity is depicted as a green arrow. The normal force is shown as a yellow arrow. The velocity of the bead is shown by a cyan arrow. The y-direction is depicted by a blue arrow, and the x-direction is depicted by a red arrow.

B. VPython Code

1. An object confined to a curve described by the hyperbolic tangent function and subjected to an electric, a magnetic, and a gravitational force.

If one would like to observe an object confined to only the gravitational force, set the objects charge, q, equal to zero.

```
from visual import *
from visual.graph import *

yaxis = arrow(pos=(0,-5,0), axis=(0,1,0), color=color.blue)
xaxis = arrow(pos=(0,-5,0), axis=(1,0,0), color=color.red)

def y(x):
    return 1 + tanh(x)
def yp(x):
    return 1/cosh(x)**2
def ypp(x):
    return -2*tanh(x)*(1/cosh(x)**2)

endpt = 5

obj = sphere(pos=(-endpt,y(-endpt),0), radius=.25, color=color.red)

c = curve( x = arange(-endpt,endpt,0.1) )
c.y = y(c.x)
nscale = .1
g = 9.8
m = 1
E = 10
B = 2
q = 1
norm = arrow(pos=obj.pos, axis=(0,0,0), color=color.yellow)
elec = arrow(pos=obj.pos, axis=(-nscale*q*E,0,0),
color=color.orange)
magn = arrow(pos=obj.pos, axis=(0,0,0), color=color.magenta)
grav = arrow(pos=obj.pos, axis=(0,-nscale*g,0), color=color.green)
v0 = 15
theta = arctan(yp(obj.pos.x))
obj.velocity = vector(v0*cos(theta),v0*sin(theta),0)
accel = vector(0,0,0)

vscale = .12
vel = arrow(pos=obj.pos, axis=vscale*obj.velocity, color=color.cyan)
gdisplay(xtitle='time', ytitle='position', x = 450 foreground =
color.black, background = color.white)
fpos_x = gdots(color=color.red, size=1)
fpos_y = gdots(color=color.blue, size=1)
gdisplay(xtitle='time', ytitle='velocity', y = 450, x = 450
foreground = color.black, background = color.white)
fvel_x = gdots(color=color.red, size=1)
fvel_y = gdots(color=color.blue, size=1)

t = 0
```

```
    dt = 0.001
    while abs(obj.pos.x) <= 5:
        rate(100)
        accel.x = (-q*E/m - q*obj.velocity.y*B/m -
    yp(obj.pos.x)*(ypp(obj.pos.x)*obj.velocity.x**2 -
    q*obj.velocity.x*B/m + g))/(1 + yp(obj.pos.x)**2)
        accel.y = ypp(obj.pos.x)*obj.velocity.x**2 +
    yp(obj.pos.x)*accel.x

        fpos_x.plot( pos=(t, obj.pos.x))
        fpos_y.plot( pos=(t, obj.pos.y))
        fvel_x.plot( pos=(t, obj.velocity.x))
        fvel_y.plot( pos=(t, obj.velocity.y))

        norm.axis=(nscale*(m*accel.x + q*E +
    q*obj.velocity.y*B),nscale*(m*(accel.y + g) - q*obj.velocity.x*B),0)
        norm.pos = obj.pos
        elec.pos = obj.pos
        magn.axis=(nscale*(-
    q*obj.velocity.y*B),nscale*(q*obj.velocity.x*B),0)
        magn.pos = obj.pos
        grav.pos = obj.pos
        vel.axis = vscale*obj.velocity
        vel.pos = obj.pos
        theta = arctan(yp(obj.pos.x))


        obj.pos = obj.pos + obj.velocity*dt
        obj.velocity = obj.velocity + accel*dt

    t = t + dt
    ##
    ##
    print(obj.velocity.x**2)
    print(obj.velocity.y**2)
    print(sqrt(obj.velocity.x**2 + obj.velocity.y**2))
```

2. An object confined to a curve described by the Trisectrix of Maclaurin and subjected to an electric, a magnetic, and a gravitational force.

If one would like to observe an object confined to only the gravitational force, set the objects charge, q, equal to zero.

```
from visual import *
from visual.graph import *

yaxis = arrow(pos=(0,-7,0), axis=(0,1,0), color=color.blue)
xaxis = arrow(pos=(0,-7,0), axis=(1,0,0), color=color.red)

R = 3

##x=a(t2-3)/(t2+1), y=at(t2-3)/(t2+1)
def y(u):
    return -R*(u**2 - 3)/(u**2 + 1)
def yp(u):
```

```
    return -8*R*u/(u**2 + 1)**2
def ypp(u):
    return 8*R*(3*u**2 - 1)/(u**2 + 1)**3

def x(u):
    return R*u*(u**2 - 3)/(u**2 + 1)
def xp(u):
    return R*(u**4 + 6*u**2 - 3)/(u**2 + 1)**2
def xpp(u):
    return -8*R*u*(u**2 - 3)/(u**2 + 1)**3

u = -2*pi
ud = 10/xp(u) #(initial velocity in x-direction)/(initial x') = initial
ud

obj = sphere(pos=(x(u),y(u),0), radius=.4, color=color.red)
obj.velocity = vector(xp(u)*ud,yp(u)*ud,0)

c = curve( u = arange(-2*pi,2*pi,0.01) )
c.y = y(c.u)
c.x = x(c.u)


yinit = y(u)

nscale = .5
g = 9.8
m = 1
E = 0 #E is positive if it is in the negative x direction
B = 0 #B is positive if it is in the negative z direction
q = 1
norm = arrow(pos=obj.pos, axis=(0,0,0), color=color.yellow)
grav = arrow(pos=obj.pos, axis=(0,-nscale*g,0), color=color.green)
elec = arrow(pos=obj.pos, axis=(-nscale*q*E,0,0), color=color.orange)
magn = arrow(pos=obj.pos, axis=(0,0,0), color=color.magenta)

print(.5*m*mag(obj.velocity)**2)

accel = vector(0,0,0)

vscale = .4
vel = arrow(pos=obj.pos, axis=vscale*obj.velocity, color=color.cyan)
gdisplay(xtitle='time', ytitle='position', x = 450, foreground =
color.black, background = color.white)
fpos_x = gdots(color=color.red, size=1)
fpos_y = gdots(color=color.blue, size=1)
gdisplay(xtitle='time', ytitle='velocity', y = 450, x = 450, foreground
= color.black, background = color.white)
fvel_x = gdots(color=color.red, size=1)
fvel_y = gdots(color=color.blue, size=1)

##gdisplay(xtitle='time', ytitle='Normal Force', y = 450, x = 450,
foreground = color.black, background = color.white)
##fvel_x = gdots(color=color.black, size=1)

t = 0
dt = 0.001
while abs(u) <= 2*pi:
```

```
    rate(300)
    udd = (-xpp(u)*ud**2 - q*yp(u)*ud*B/m - q*E/m -
(yp(u)/xp(u))*(ypp(u)*ud**2 - q*xp(u)*ud*B/m + g) )/(xp(u) +
(yp(u)**2)/xp(u))
    accel.x = xpp(u)*ud**2 + xp(u)*udd
    accel.y = ypp(u)*ud**2 + yp(u)*udd

    fpos_x.plot( pos=(t, obj.pos.x))
    fpos_y.plot( pos=(t, obj.pos.y))
    fvel_x.plot( pos=(t, obj.velocity.x))
    fvel_y.plot( pos=(t, obj.velocity.y))

    norm.axis=(nscale*(m*accel.x + q*E +
q*obj.velocity.y*B),nscale*(m*(accel.y + g) - q*obj.velocity.x*B),0)
    norm.pos = obj.pos
    magn.axis=(nscale*(-
q*obj.velocity.y*B),nscale*(q*obj.velocity.x*B),0)
    magn.pos = obj.pos
    elec.pos = obj.pos
    grav.pos = obj.pos
    vel.axis = vscale*obj.velocity
    vel.pos = obj.pos

    obj.pos = obj.pos + obj.velocity*dt
    obj.velocity = obj.velocity + accel*dt

##    fvel_x.plot( pos=(t,sqrt(norm.axis.x**2 + norm.axis.y**2)))

    u = u + ud*dt
    ud = ud + udd*dt
    t = t + dt
print(m*g*(y(u)-yinit) + .5*m*mag(obj.velocity)**2)
print(t)
```

3. An object confined to a curve described by the hyperbolic tangent function and subjected to kinetic and static friction, and a gravitational force.

Incomplete: Does not successfully describe the situation where the bead traverses the wire backwards after coming to a stop and not having static friction affect the object.

```
from visual import *
from visual.graph import *

yaxis = arrow(pos=(0,-5,0), axis=(0,1,0), color=color.blue)
xaxis = arrow(pos=(0,-5,0), axis=(1,0,0), color=color.red)

def y(x):
    return 1 + tanh(x)
def yp(x):
    return 1/cosh(x)**2
def ypp(x):
    return -2*tanh(x)*(1/cosh(x)**2)

endpt = 5
```

```
obj = sphere(pos=(-endpt,y(-endpt),0), radius=.25, color=color.red)
c = curve( x = arange(-endpt,endpt,0.1) )
c.y = y(c.x)
nscale = .1
g = 9.8
m = 1
mu = 1
mus = 10
norm = arrow(pos=obj.pos, axis=(0,0,0), color=color.yellow)
fric = arrow(pos=obj.pos, axis=(0,0,0), color=color.orange)
grav = arrow(pos=obj.pos, axis=(0,-nscale*g,0), color=color.green)
v0 = 18
theta = arctan(yp(obj.pos.x))
obj.velocity = vector(v0*cos(theta),v0*sin(theta),0)
accel = vector(0,0,0)

vscale = .2
vel = arrow(pos=obj.pos, axis=vscale*obj.velocity, color=color.cyan)
gdisplay(xtitle='time', ytitle='position', x = 450, foreground =
color.black, background = color.white)
fpos_x = gdots(color=color.red, size=1)
fpos_y = gdots(color=color.blue, size=1)
gdisplay(xtitle='time', ytitle='velocity', y = 450, x = 450, foreground
= color.black, background = color.white)
fvel_x = gdots(color=color.red, size=1)
fvel_y = gdots(color=color.blue, size=1)

oldvel = obj.velocity

t = 0
dt = 0.001
while abs(obj.pos.x) <= 5 and dt > 0:

    rate(100)
    gamma = (-sin(theta) - mu*cos(theta))/(cos(theta) - mu*sin(theta))
    accel.x = (ypp(obj.pos.x)*obj.velocity.x**2 + g)*gamma/(1 -
yp(obj.pos.x)*gamma)
    accel.y = ypp(obj.pos.x)*obj.velocity.x**2 + yp(obj.pos.x)*accel.x

    fpos_x.plot( pos=(t, obj.pos.x))
    fpos_y.plot( pos=(t, obj.pos.y))
    fvel_x.plot( pos=(t, obj.velocity.x))
    fvel_y.plot( pos=(t, obj.velocity.y))

    norm.axis=(-
nscale*m*g*cos(theta)*sin(theta),nscale*m*g*cos(theta)*cos(theta),0)
    ## assumeing v0 doesnt = 0 we will always start with kinetic
friction

    fric.axis=(-
obj.velocity.x/abs(obj.velocity.x)*mu*sqrt(norm.axis.x**2 +
norm.axis.y**2)*cos(theta),-
obj.velocity.x/abs(obj.velocity.x)*mu*sqrt(norm.axis.x**2 +
norm.axis.y**2)*sin(theta),0)
        #fric's nscale is within the mag of norm

    if obj.velocity.dot(oldvel) < 0 and tan(theta) <=  mus:
        dt = 0
```

```
    norm.pos = obj.pos
    grav.pos = obj.pos
    fric.pos = obj.pos
    vel.axis = vscale*obj.velocity
    vel.pos = obj.pos
    theta = arctan(yp(obj.pos.x))
    oldvel = obj.velocity
    obj.pos = obj.pos + obj.velocity*dt
    obj.velocity = obj.velocity + accel*dt

    t = t + dt
```

4. An object confined to a curve described by the Trisectrix of Maclaurin and subjected to kinetic and static friction, and a gravitational force.

Incomplete: Does not successfully describe the situation where the bead traverses the wire backwards after coming to a stop and not having static friction affect the object.

```
from visual import *
from visual.graph import *

yaxis = arrow(pos=(0,-5,0), axis=(0,1,0), color=color.blue)
xaxis = arrow(pos=(0,-5,0), axis=(1,0,0), color=color.red)

R = 3

##x=a(t2-3)/(t2+1), y=at(t2-3)/(t2+1)
def y(u):
    return -R*(u**2 - 3)/(u**2 + 1)
def yp(u):
    return -8*R*u/(u**2 + 1)**2
def ypp(u):
    return 8*R*(3*u**2 - 1)/(u**2 + 1)**3

def x(u):
    return R*u*(u**2 - 3)/(u**2 + 1)
def xp(u):
    return R*(u**4 + 6*u**2 - 3)/(u**2 + 1)**2
def xpp(u):
    return -8*R*u*(u**2 - 3)/(u**2 + 1)**3

u = -2*pi
ud = 65.64/xp(u) #(initial velocity in x-direction)/(initial x') =
initial ud

obj = sphere(pos=(x(u),y(u),0), radius=.25, color=color.red)
obj.velocity = vector(xp(u)*ud,yp(u)*ud,0)

c = curve( u = arange(-2*pi,2*pi,0.01) )
c.y = y(c.u)
c.x = x(c.u)
theta = arctan(yp(u)/xp(u))
nscale = .5
g = 9.8
```

```
m = 1
mu = .8
mus = 13
norm = arrow(pos=obj.pos, axis=(0,0,0), color=color.yellow)
grav = arrow(pos=obj.pos, axis=(0,-nscale*g,0), color=color.green)
fric = arrow(pos=obj.pos, axis=(0,0,0), color=color.orange)

accel = vector(0,0,0)

vscale = .12
vel = arrow(pos=obj.pos, axis=vscale*obj.velocity, color=color.cyan)
gdisplay(xtitle='time', ytitle='position', x = 450, foreground =
color.black, background = color.white)
fpos_x = gdots(color=color.red, size=1)
fpos_y = gdots(color=color.blue, size=1)
gdisplay(xtitle='time', ytitle='velocity', y = 450, x = 450, foreground
= color.black, background = color.white)
fvel_x = gdots(color=color.red, size=1)
fvel_y = gdots(color=color.blue, size=1)

oldud = ud

t = 0
dt = 0.001
while abs(u) <= 2*pi and dt > 0 :
    rate(100)

    udd = ((ypp(u)*ud**2 + g)*((-sin(theta) -
mu*cos(theta))/(cos(theta) - mu*sin(theta))) - xpp(u)*ud**2)/(xp(u) -
((-sin(theta) - mu*cos(theta))/(cos(theta) - mu*sin(theta)))*yp(u))
    accel.x = xpp(u)*ud**2 + xp(u)*udd
    accel.y = ypp(u)*ud**2 + yp(u)*udd

    fpos_x.plot( pos=(t, obj.pos.x))
    fpos_y.plot( pos=(t, obj.pos.y))
    fvel_x.plot( pos=(t, obj.velocity.x))
    fvel_y.plot( pos=(t, obj.velocity.y))

    if ud/abs(ud) + oldud/abs(oldud) == 0 and abs(tan(theta)) <=  mus:
        dt = 0

    norm.axis=(-
nscale*m*g*cos(theta)*sin(theta),nscale*m*g*cos(theta)*cos(theta),0)
    fric.axis=(-
obj.velocity.x/abs(obj.velocity.x)*mu*sqrt(norm.axis.x**2 +
norm.axis.y**2)*cos(theta),-
obj.velocity.x/abs(obj.velocity.x)*mu*sqrt(norm.axis.x**2 +
norm.axis.y**2)*sin(theta),0)
    norm.pos = obj.pos
    fric.pos = obj.pos
    grav.pos = obj.pos
    vel.axis = vscale*obj.velocity
    vel.pos = obj.pos

    theta = arctan(yp(u)/xp(u))

    obj.pos = obj.pos + obj.velocity*dt
    obj.velocity = obj.velocity + accel*dt
```

```
    oldud = ud

    u = u + ud*dt
    ud = ud + udd*dt

    t = t + dt

##
##
print(obj.velocity.x)
print(obj.velocity.y)
print(sqrt(obj.velocity.x**2 + obj.velocity.y**2))
```

5. An object confined to a curve described by the hyperbolic tangent function and subjected to quadratic and linear drag, and the gravitational force.

Incomplete: Does not successfully describe the situation where the bead traverses the wire backwards after coming to a stop at some location on the incline.

```
from visual import *
from visual.graph import *

yaxis = arrow(pos=(0,-5,0), axis=(0,1,0), color=color.blue)
xaxis = arrow(pos=(0,-5,0), axis=(1,0,0), color=color.red)

def y(x):
    return 1 + tanh(x)
def yp(x):
    return 1/cosh(x)**2
def ypp(x):
    return -2*tanh(x)*(1/cosh(x)**2)

endpt = 5

obj = sphere(pos=(-endpt,y(-endpt),0), radius=.25, color=color.red)

c = curve( x = arange(-endpt,endpt,0.1) )
c.y = y(c.x)


nscale = .1
g = 9.8
m = 1
b = 10
d = 1
norm = arrow(pos=obj.pos, axis=(0,0,0), color=color.yellow)
ldrg = arrow(pos=obj.pos, axis=(0,0,0), color=color.orange)
qdrg = arrow(pos=obj.pos, axis=(0,0,0), color=color.magenta)
grav = arrow(pos=obj.pos, axis=(0,-nscale*g,0), color=color.green)
v0 = 70
theta = arctan(yp(obj.pos.x))
obj.velocity = vector(v0*cos(theta),v0*sin(theta),0)
accel = vector(0,0,0)

vscale = .2
vel = arrow(pos=obj.pos, axis=vscale*obj.velocity, color=color.cyan)
```

```
gdisplay(xtitle='time', ytitle='position', x = 450)
fpos_x = gdots(color=color.red, size=1)
fpos_y = gdots(color=color.blue, size=1)
gdisplay(xtitle='time', ytitle='velocity', y = 450, x = 450)
fvel_x = gdots(color=color.red, size=1)
fvel_y = gdots(color=color.blue, size=1)

t = 0
dt = 0.001
while abs(obj.pos.x) <= 5:
    rate(100)
    accel.x = (-yp(obj.pos.x)*(ypp(obj.pos.x)*obj.velocity.x**2 +
b/m*obj.velocity.y + d/m*obj.velocity.y**2 + g) - b/m*obj.velocity.x -
d/m*obj.velocity.x**2)/(1 + yp(obj.pos.x)**2)
    accel.y = ypp(obj.pos.x)*obj.velocity.x**2 + yp(obj.pos.x)*accel.x

    fpos_x.plot( pos=(t, obj.pos.x))
    fpos_y.plot( pos=(t, obj.pos.y))
    fvel_x.plot( pos=(t, obj.velocity.x))
    fvel_y.plot( pos=(t, obj.velocity.y))

    norm.axis=(nscale*(m*accel.x + b*obj.velocity.x +
d*obj.velocity.x**2),nscale*(m*(accel.y + g) + b*obj.velocity.y +
d*obj.velocity.y**2),0)
    ldrg.axis=(nscale*(-b*obj.velocity.x),nscale*(-b*obj.velocity.y),0)
    qdrg.axis=(nscale*(-d*obj.velocity.x**2),nscale*(-
d*obj.velocity.y**2),0)
    norm.pos = obj.pos
    grav.pos = obj.pos
    ldrg.pos = obj.pos
    qdrg.pos = obj.pos
    vel.axis = vscale*obj.velocity
    vel.pos = obj.pos
    theta = arctan(yp(obj.pos.x))

    obj.pos = obj.pos + obj.velocity*dt
    obj.velocity = obj.velocity + accel*dt

    t = t + dt
##
##
print(obj.velocity.x**2)
print(obj.velocity.y**2)
print(sqrt(obj.velocity.x**2 + obj.velocity.y**2))
```